

## **REMARKS**

### **Claim Amendments**

Claims 1-34 are pending, including independent claims 1, 13, 21, 24, 33 and 34.

Applicants have amended independent claims 1, 33 and 34 to incorporate the features recited in claims 7 and 8. Independent claim 13 has been amended to incorporate the features recited in claims 14 and 15. And, independent claims 21 and 24 have been amended to incorporate the features recited in claims 26 and 27.

In view of the above amendments to the independent claims, Applicants have canceled claims 2-5, 7, 8, 14-16, 22 and 25-27.

The dependencies of claims 6, 9, 17 and 28 have been amended in view of the above amendments.

No new matter or new issues have been introduced into the claims by these amendments.

### **35 U.S.C. 103 Rejection**

The Examiner has rejected claims 1, 3, 4, 7, 13, 14, 21, 23, 24, 26, 33 and 34 under 35 U.S.C. 103(a), stating that these claims are unpatentable over Kannan et al. (US Patent No. 5,818,702, hereinafter called "Kannan") in view of Bak et al. (US Patent No. 6,415,381 B1, hereinafter called "Bak"). Applicants respectfully request reconsideration of the rejection based on the amended claims for the reasons set out below.

### **Claim 1**

Amended claim 1 recites a method for recovering an application from a runtime fault. The method receives an exception caused due to a runtime fault in a thread executing a C++ based application, and traps the exception before the exception reaches a top level exception handler. The method translates the trapped exception into a C++ exception that the C++

application is capable of handling. The trapped exception is translated into the C++ exception which is able to be resolved by an application defined C++ exception handler. The method determines if there is an application based C++ exception handler which is capable of resolving the translated exception, terminates the thread that caused the exception when there is no C++ based exception handler which is capable of resolving the translated exception, and continues execution of the application.

In contrast, none of Kannan, Bak or Anschuetz disclose or suggest such a method for recovering an application from a runtime fault.

As discussed in Amendments A-C, which are incorporated by reference, Kannan replaces existing offending message loop with a safe message loop 109 that is pre-stored in the memory 103 in advance (column 6, lines 36-37). Replacing the existing message loop with the pre-stored safe message loop 109 may not always work as if nothing happened. There may be some logic around the existing message loop that, if replaced with the safe message loop 109, may lead to undesired behaviour of the application. Kannan does not teach or suggest translating the trapped exception into a C++ exception, or terminating a thread that caused the exception.

Bak transforms an exception into a different programming language for propagating it in an execution stack storing frames for functions in multiple programming languages (column 11, lines 25-32). Bak does not teach or suggest translating the trapped exception into a C++ exception. Even if the concept of transforming an exception into a different programming languages was applied to translation of an exception into a C++ exception, Bak stil fails to teach or suggest termination of a thread that caused the exception.

Amended claim 1 now recites the limitations originally recited in claims 7 and 8. The Examiner has rejected claim 8 based on Kannan in view of Bak further in view of Anschuetz et al. (US Patent 5,305,455, hereinafter called "Anschuetz").

Anschuetz deals with handing of an operating system level exception. Anschuetz does not disclose exception handing at the application level. Anschuetz states that "While an operating system is a collection of programs that provide many different functions, the general function to which the invention pertains is that of task management and, more specifically, exception management. ... The general function of task management is to allocate resources to the different programs so that the various programs can be concurrently run in different time slices" (column 1, lines 22-34). Anschuetz receives an exception of an operation system, and dispatches the exception in the operation system to an exception handler.

The thread referred to in Anschuetz, e.g., column 5, lines 9-16, is a thread executing a process in the operation system. Anschuetz's system manages exceptions on a thread basis in the operating system. As the thread in the operating system is managing the resources allocation, terminating a thread in the operation system level would allow the process to run when a resources allocation problem occurs. However, Anschuetz's system is not able to handle other exceptions that are caused due to an application's runtime faults. Terminating a thread at the operation system level does not allow the application to handle its exceptions programmatically.

In contrast, the thread recited in amended claim 1 is "a thread executing a C++ based application". The method as recited in amended claim 1 allows handling of such a thread based on an application defined C++ exception handler. The method terminates such a thread that was executing the C++ based application and caused the exception if there is no application defined C++ exception handler that is capable of resolving the exception. To allow this

exception handling, the method translates the trapped exception into a C++ exception that the C++ based application is capable of handling. Thus, the present invention as recited in amended claim 1 enables the C++ based application to handle exceptions by itself programmatically.

Anschuetz does not teach or suggest any method of handling exceptions caused due to a runtime fault in a thread executing an application, or translating an exception to a C++ exception to allow the application to handle the exception.

None of Kannan, Bak nor Anschuetz teaches or suggests translating the trapped exception into a C++ exception, or terminating a thread that was executing an C++ based application and caused the exception when there is no C++ based exception handler which is capable of resolving the translated exception.

Accordingly, even if one skilled in the art were to trap an execution as per the teaching of Kannan and transforms it into a different language as per the teaching of Bak, he would still fail to achieve the method as recited in amended claim 1. He would simply replace the transformed exception with a safe message loop as per the teaching of Kannan, or terminates the thread executing the process in the operation system level as per the teaching of Anschuetz. If there is an exception stack, he would propagate the transformed exception into the exception stack as per the teaching of Bak. He would however still be unable to terminate a thread that was executing a C++ based application and caused the exception when there is no C++ based exception handler which is capable of resolving the translated exception. He would not be able to allow the application to handle its exceptions.

Therefore, Applicants respectfully submit that the invention as recited in amended claim 1 is unobvious and has been patentably distinguished over the combination of Kannan, Bak and Anschuetz.

**JAYES SOLOWAY P.C.**  
30 W. CUSHING STREET  
TUCSON, AZ 85701  
TEL. 520.882.7623  
FAX. 520.882.7643

175 CANAL STREET  
MANCHESTER, NH 03101  
TEL. 603.668.1400  
FAX. 603.668.8567

**Other independent claims 13, 21, 24, 33 and 34**

Other independent claims 13, 21, 24, 33 and 34 as amended also recite the steps or corresponding elements as recited in amended claim 1. Thus, the above arguments also apply to these claims.

Accordingly, Applicants respectfully submit that these claims are also unobvious and have been patentably distinguished over the combination of Kannan, Bak and Anschuetz.

**Dependent claims 3, 4, 7, 14, 23 and 26**

Dependent claims 3, 4, 7, 14, 23 and 26 have been canceled.

**35 U.S.C. 103(a) Rejection of Claims 2, 5, 6, 8, 15, 16, 22, 25, 27 and 28**

The Examiner has rejected claims 2, 5, 6, 8, 15, 16, 22, 25, 27 and 28 under 35 U.S.C. 103(a), stating that these claims are unpatentable over Kannan in view of Bak further in view of Anschuetz et al.

Claims 2, 5, 8, 15, 16, 22, 25 and 27 have been canceled.

Claims 6 and 28 have been amended to depend on claims 1 and 24, respectively.

As discussed above, claims 1 and 24 are unobvious over Kannan and Bak and Anschuetz. Accordingly, it is respectfully submitted that amended claims 6 and 28 have been also patentably distinguished over Kannan and Bak and Anschuetz.

**35 U.S.C. 103(a) Rejection of Claims 9-12, 17-20 and 29**

The Examiner has rejected claims 9-12, 17-20 and 29 under 35 U.S.C. 103(a), stating that these claims are unpatentable over Kannan in view of Bak further in view of Anschuetz further in view of LeVine et al. (US Patent 6,591,379 B1, hereinafter called "LeVine").

Applicants respectfully request reconsideration of this rejection for the reasons set out below.

These claims directly or indirectly depend on amended claims 1, 13 and 24.

Accordingly, it is respectfully submitted that claims 9-12, 17-20 and 29 have been also patentably distinguished over Kannan and Bak and Anschuetz further in view of LeVine.

**35 U.S.C. 103(a) Rejection of Claims 30-32**

The Examiner has rejected claims 30-32 under 35 U.S.C. 103(a), stating that these claims are unpatentable over Kannan in view of Bak further in view of Anschuetz further in view of LeVine further in view of Lillevold (US Patent 6,230,284 B1, hereinafter called "Lillevold"). Applicants respectfully request reconsideration of this rejection for the reasons set out below.

These claims directly or indirectly depend on amended claim 24. Accordingly, it is respectfully submitted that claims 30-32 have been also patentably distinguished over Kannan and Bak and Anschuetz further in view of LeVine and Lillevold.

Consequently, it is respectfully submitted that claims 1, 6, 9-13, 17-21, 23, 24 and 28-34 as amended, which are currently on file, comply with the requirements under 35 U.S.C. 103.

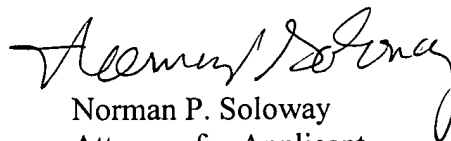
Applicants trust that the application is now in a condition for allowance.

Reconsideration of the application is respectfully requested.

Having dealt with all the objections raised by the Examiner, the Application is believed to be in order for allowance. Early and favorable action are respectfully requested.

In the event there are any fee deficiencies or additional fees are payable, please charge them (or credit any overpayment) to our Deposit Account Number 08-1391.

Respectfully submitted,



Norman P. Soloway  
Attorney for Applicant  
Reg. No. 24,315

1AYES SOLOWAY P.C.  
30 W. CUSHING STREET  
TUCSON, AZ 85701  
TEL. 520.882.7623  
FAX. 520.882.7643

175 CANAL STREET  
1ANCHESTER, NH 03101  
TEL. 603.668.1400  
FAX. 603.668.8567



CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail in an envelope addressed to: MAIL STOP RCE, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on March 15, 2006, at Tucson, Arizona.

NPS:cg/kmg

By 

1AYES SOLOWAY P.C.  
30 W. CUSHING STREET  
TUCSON, AZ 85701  
TEL. 520.882.7623  
FAX. 520.882.7643

175 CANAL STREET  
1ANCHESTER, NH 03101  
TEL. 603.668.1400  
FAX. 603.668.8567